

Sisteme de Prelucrare Grafică

Curs nr. 9

4.3 Modele de redare a suprafețelor iluminate (modele de umbrire)

Aplicarea directă a relațiilor de calcul a intensității (culorii) fiecărui pixel necesită un timp de execuție prea mare. Din acest motiv, în sinteza de imagini se folosesc uneori metode simplificate de calcul a culorii fiecărui pixel, numite *tehnici sau modele de umbrire*. Pentru obiectele modelate prin rețea de poligoane se folosesc următoarele modele de umbrire:

- umbrire constantă;
- umbrire Gouraud;
- umbrire Phong.

4.3.1 Modelul de umbrire constantă

Pentru fiecare suprafață se calculează o singură intensitate a culorii. Acest mod de calcul presupune că sursele de lumină și observatorul sunt la infinit. Calculul umbririi constante înseamnă aplicarea formulelor Phong pentru fiecare suprafață vizibilă, o singură dată, pentru fiecare poziție a observatorului.

4.3.2 Modelul de umbrire Gouraud

În acest model se calculează o intensitate în fiecare vârf al suprafeței pe baza relațiilor din modelul de reflexie Phong. Intensitățile în toate celelalte puncte ale suprafeței se obțin prin interpolare liniară a intensităților din vârfuri, pe parcursul generării interiorului poligonului.

Procesul de redare a unei suprafețe iluminate după modelul Gouraud cuprinde următoarele etape:

1. - se calculează normala în fiecare vârf al suprafeței, ca medie aritmetică a normalelor tuturor poligoanelor adiacente vârfurilor (fig. 1)

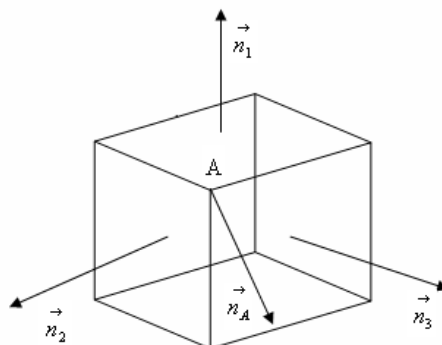


Fig. 1

$$\vec{n}_v = \sum_{i=0}^n \frac{\vec{n}_i}{m}. \quad (1)$$

$$\vec{n}_A = \frac{\vec{n}_1 + \vec{n}_2 + \vec{n}_3}{3}.$$

2. - se calculează o intensitate în fiecare vârf, considerându-se ca normala la suprafață, normala vârfului

3. - se afișează suprafața calculând intensitatea într-un punct al poligonului astfel:

- a) pentru punctele de pe laturi prin interpolarea liniară a intensității vârfurilor;
- b) pentru punctele interioare se interpolează intensitățile de pe laturi.

De exemplu, considerăm poligonul din fig. 2

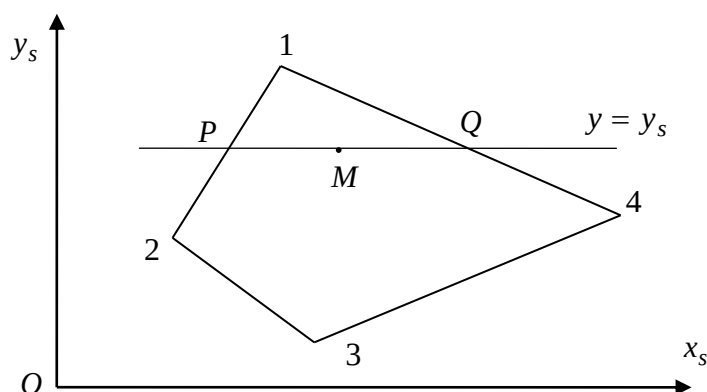


Fig. 2

Intensitatea punctelor de pe linia $y = y_s$ se calculează astfel:

$$\begin{aligned} I_P &= I_1 - \frac{I_1 - I_2}{y_1 - y_2} (y_1 - y), \\ I_Q &= I_1 - \frac{I_1 - I_4}{y_1 - y_4} (y_1 - y), \\ I_M &= I_P - \frac{I_P - I_Q}{x_P - x_Q} (x_P - x_M). \end{aligned} \quad (2)$$

Avantaje: modelul Gouraud se poate integra ușor în algoritmii de generare a scenelor 3D, de exemplu în Z-buffer sau scan-line.

Dezavantaje:

- nu se poate obține o pată de lumină (highlight) datorată reflexiei speculare în interiorul unui poligon dacă vârfurile poligonului nu sunt în această pată de lumină. (Intensitatea punctelor interioare nu poate fi mai mare decât a vârfurilor deoarece se obține prin interpolarea lor).

- nu elimină complet discontinuitățile intensității luminoase la trecerea de la o suprafață la alta.

4.3.3. Modelul de umbrire Phong

O parte a deficiențelor modelului Gouraud sunt eliminate în modelul Phong. În acest model se calculează prin interpolare normală în fiecare punct al suprafeței de vizualizat. Pentru afișarea suprafeței se parcurg următoarele etape:

1. - se determină o normală în fiecare vârf al suprafeței ca și în modelul Gouraud;
2. - se afișează suprafața calculând în fiecare punct vizibil mai întâi normala apoi intensitatea luminoasă. Calculul normalelor se face prin interpolare și este similar cu calculul intensităților la modelul Gouraud .

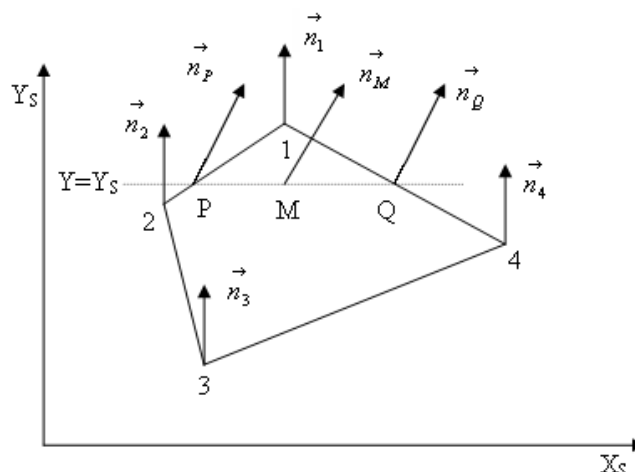


Fig. 3

$$\begin{aligned}
 \vec{n}_P &= \vec{n}_1 - \frac{\vec{n}_1 - \vec{n}_2}{y_1 - y_2} (y_1 - y), \\
 \vec{n}_Q &= \vec{n}_1 - \frac{\vec{n}_1 - \vec{n}_4}{y_1 - y_4} (y_1 - y), \\
 \vec{n}_M &= \vec{n}_P - \frac{\vec{n}_P - \vec{n}_Q}{x_P - x_Q} (x_P - x_M).
 \end{aligned} \tag{3}$$

Modelul Phong permite considerarea componentei speculare în calculul intensității tuturor punctelor suprafeței. Din punct de vedere al operațiilor efectuate, umbrirea Phong necesită mai multe calcule decât modelul Gouraud, deoarece pentru fiecare pixel se calculează intensitatea luminoasă. Calitatea imaginilor este mult mai bună decât în modelul Gouraud.

Modelele Gouraud și Phong sunt incluse în standardele grafice GKS3D și PHIGS (modele implementate hardware). În OpenGL este inclus numai modelul Gouraud .

4.4 Generarea fenomenelor naturale

Pentru simularea efectelor atmosferice precum *ceață*, *păclă*, *poluare atmosferică* modalitatea de vizualizare cea mai frecvent folosită constă în modificarea culorilor obiectelor din scenă pe baza distanței față de punctul de observare. Ceața (ca și păcla sau poluarea atmosferică) este simulată prin combinarea culorii ceții cu culoarea fragmentelor suprafețelor,

folosind un factor de combinare care depinde de distanța acestora față de punctul de observare. Un *fragment* al unei suprafețe este porțiunea din suprafață vizibilă într-un pixel.

În simularea ceții se folosesc mai multe formule de calcul al factorului f de combinare a culorilor :

$$f = e^{-dens \cdot z_s}, \quad (4)$$

$$f = e^{-(dens \cdot z_s)^2}, \quad (5)$$

$$f = \frac{end - z_s}{end - start}. \quad (6)$$

Observație: $0 \leq f \leq 1$.

În primele două formule factorul de combinare variază exponențial iar parametrul *dens* (un număr pozitiv) poate fi variat pentru obținerea unor efecte diferite de ceață .

În a treia relație, f variază liniar de la punctul de observare între limitele *start* (de unde începe ceața) și *end* (unde se termină) .

Fiecare componentă C_r a culorii unui fragment (C poate fi R, G, B, A) se calculează prin combinarea culorii C_s a fragmentului cu culoarea C_f a ceții, prin ecuația:

$$C_r = f C_s + (1 - f) C_f. \quad (7)$$

În cazul utilizării unui factor liniar, trebuie avut în vedere că la limita maximă $z_s = end$ $f = 0$ și deci nu se mai distinge nici un obiect. Deci *end* trebuie corelată cu distanța la planul de vizibilitate depărtat (*far*)

$$end \leq far. \quad (8)$$

4.5. Transparența

Atunci când lumina trece dintr-un mediu în altul direcția sa se modifică datorită refracției (fig. 4):

$$n_1 \sin i = n_2 \sin r \quad (9)$$

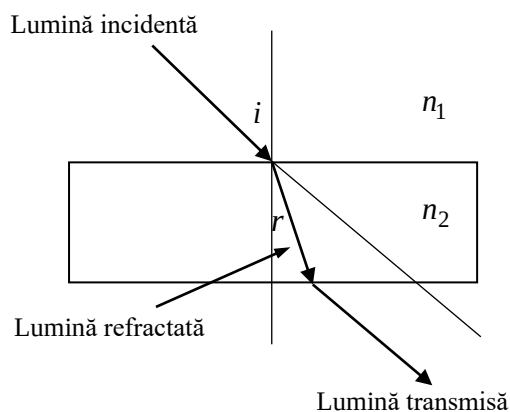


Fig. 4

Refracția produce o distorsiune a obiectelor. Multe metode practice de modelare a transparenței ignoră refracția. În acest fel, obiectele vizibile sunt cele aflate pe direcția razei incidente. De obicei motivul ignorării refracției nu-l constituie calculul suplimentar, ci faptul că astfel obiectele nu apar deformate.

Atunci când suprafața vizibilă este transparentă, culoare pixelului care se afișează se calculează din culoarea obiectului vizibil și celui din spatele lui .

$$I_{\lambda} = (1 - k_{t_1}) I_{\lambda_1} + k_{t_1} \cdot I_{\lambda_2} \quad (10)$$

unde:

k_{t_1} – coeficientul de transparență al obiectului 1, I_{λ_1} - culoarea obiectului 1, I_{λ_2} - culoarea obiectului 2.

$k_{t_1} \in [0,1] \Rightarrow$ dacă $k_{t_1} = 1$ - obiectul 1 este perfect transparent $\Rightarrow I_{\lambda} = I_{\lambda_2}$,
 dacă $k_{t_1} = 0$ - obiectul 1 este opac $\Rightarrow I_{\lambda} = I_{\lambda_1}$.

Dacă suprafața din spatele său e tot transparentă, se aplică recursiv formula până se ajunge la un obiect opac sau raza refractată nu întâlnește alt obiect.

4.6. Algoritmul Ray Tracing („traseul razei”)

Algoritmul Ray Tracing redă realist imaginile prin efecte de iluminare și umbră. Algoritmul calculează culoarea unui punct al unei suprafețe ținând cont de *iluminarea locală* și *iluminarea globală*. Iluminarea locală este dată de lumina care vine direct de la sursele de lumină. Iluminarea globală într-un punct este dată de lumina reflectată sau transmisă de alte obiecte în acel punct.

Algoritmul generează imaginea în funcție de poziția observatorului, urmărind traseul unor raze de lumină imaginare care pornesc de la observator la obiectele scenei.

Algoritmul discretizează planul de proiecție într-o grilă de zone dreptunghiulare corespunzătoare rezoluției ecranului.

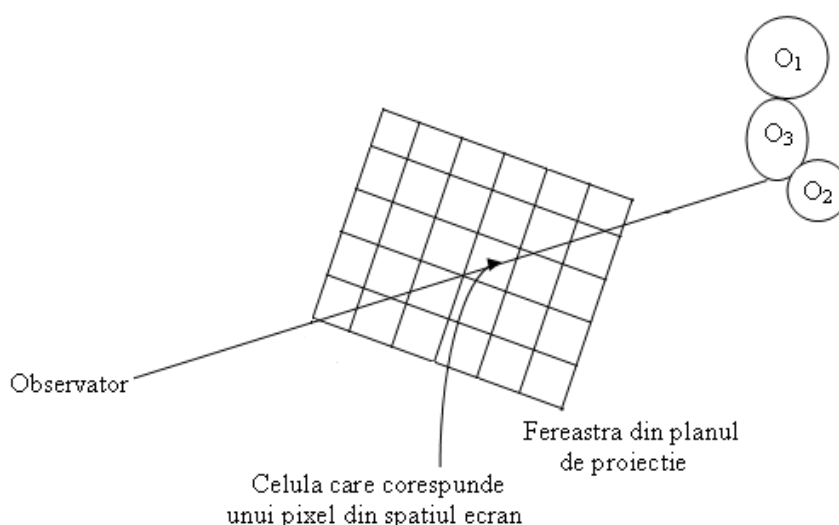


Fig. 5

Pentru fiecare pixel se consideră o rază de lumină care pornește de la observator către centrul zonei dreptunghiulare (din planul de proiecție) corespunzătoare acelui pixel și care intersectează obiectele scenei (traseul razei în sens invers).

Obiectul intersectat de rază, cel mai aproape de observator dă culoarea pixelului (algoritmul ray tracing simplu).

Algoritmul Ray Tracing simplu poate fi descris astfel :

FOR fiecare pixel DO

- determină raza din centrul de proiecție în celula corespunzătoare acelui pixel;
- determină intersecția razei cu cel mai apropiat obiect;
- scrie în pixel culoarea primului obiect intersectat, dacă există, dacă nu, scrie culoarea fundalului;

ENDFOR

Acest algoritm simplu se îmbunătățește astfel: culoarea unui pixel este dată și de razele reflectate și transmise de celelalte obiecte. Un algoritm Ray Tracing complex continuă traseul razei din punctul de intersecție cel mai apropiat până când raza iese din scenă sau întâlnește un obiect opac.

Raza care pleacă de la observator se numește *rază primară*. Razele care pornesc din punctul de intersecție sunt *raze secundare*. Acestea pot fi: *raze de iluminare* (date direct de surse în punctul respectiv), *raza reflectată* și *raza refractată*. Pentru razele reflectate și refractate se continuă recursiv traseul razei, considerând că din punctul de intersecție al acestor raze cu obiectele scenei pornesc alte raze secundare.

Pentru exemplificare considerăm o scenă formată din 4 obiecte și 2 surse de lumină. Obiectele 1 și 2 le considerăm transparente, iar 3 și 4 opace .

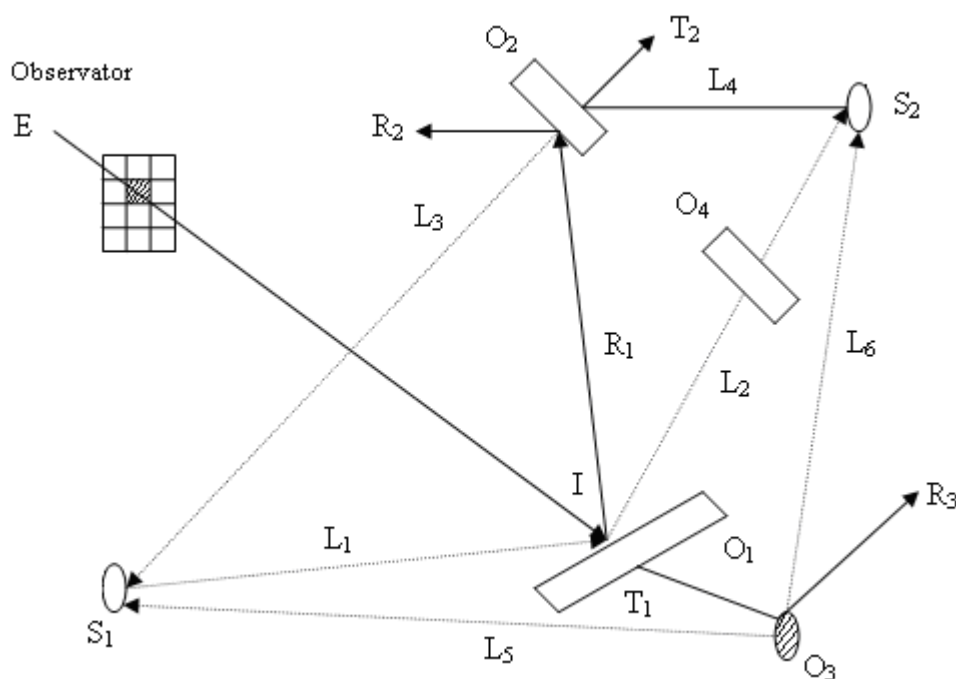


Fig. 6

Raza primară E intersectează obiectul O_1 în punctul I . În punctul I intensitatea luminoasă este dată de iluminarea locală (raza de iluminare L_1 de la sursa S_1 , raza L_2 fiind ignorată deoarece O_4 este opac) și de iluminarea globală (razele R_1 și T_1).

Urmărind în continuare traseul razei R_1 , aceasta întâlnește O_2 care este transparent. Din punctul de intersecție al lui R_1 cu O_2 se consideră că pornesc razele de iluminare L_3 și L_4 , raza reflectată R_2 și raza refractată T_2 . Raza T_2 părăsește scena, intersectând mediul înconjurător care are culoarea de fond.

Urmărind traseul razei T_1 , aceasta întâlnește obiectul O_3 care este opac. Din acest punct de intersecție pleacă razele de iluminare L_5 , L_6 și raza reflectată R_3 .

În continuare se consideră recursiv traseul razelor reflectate și refractate până când o rază părăsește scena sau contribuția sa la culoarea pixelului este prea mică. De exemplu, dacă se consideră că obiectul O_1 este 30% reflectiv, iar O_2 50% reflectiv, rezultă ca R_2 contribuie la culoarea lui I cu 15%.

Razele considerate pentru un pixel pot fi reprezentate printr-o structură de date de tip arbore (fig. 7), care este evaluat de la frunze către rădăcină. Intensitatea fiecărui nod este determinată pe baza intensității fiilor. În cazul considerat avem:

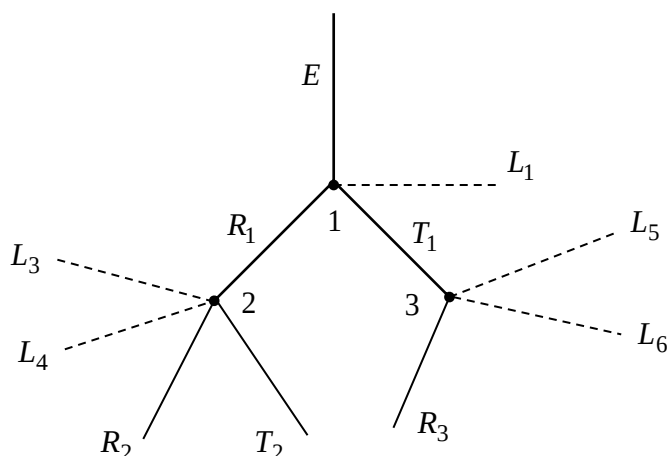


Fig. 7

Intensitatea unui pixel poate fi calculată folosind ecuația de iluminare propusă de Whitted:

$$I = I_{ambient} + I_{surse} + I_{reflectat} + I_{refractat} \quad (11)$$

$$I = k_a \cdot o_d \cdot I_a + \sum_{i=1}^m s_i \cdot f_{at,i} \cdot I_i \cdot \left[k_d \cdot o_d \cdot (\vec{l}_i \cdot \vec{n}) + k_s \cdot o_s \cdot (\vec{r}_i \cdot \vec{n})^n \right] + k_s \cdot I_r + k_t \cdot I_t \quad (12)$$

unde :

I_a - intensitatea luminii ambientale

k_a - coeficientul de difuzie a luminii ambientale

o_d - culoarea difuză a obiectului

k_s - coeficientul de reflexie speculară

o_s - culoarea speculară a obiectului

$s_i = 0$ - dacă punctul respectiv nu primește direct lumină de la sursa i

$s_i = 1$ - dacă punctul respectiv primește direct lumină de la sursa i

$f_{at,i}$ - factor de atenuare a luminii provenind de la sursa i

I_r - intensitatea luminii reflectate

I_t - intensitatea luminii transmise (prin refracție).

Obs. Algoritmul creează câte un arbore pentru fiecare pixel. Memorarea arborilor pentru toți pixelii permite modificarea proprietăților fizice ale suprafețelor și afișarea imediată a noii imagini, deoarece punctele de intersecție nu trebuie calculate (pentru un observator fix).

Aprecieri ale algoritmului Ray Tracing:

- produce imagini foto-realiste deoarece ține cont de foarte multe fenomene optice;
- este un algoritm simplu, singurele calcule geometrice pe care le presupune fiind intersecția dintre o dreaptă și obiectele scenei.

Obs. Algoritmul prezentat consideră câte o rază pentru fiecare pixel. Calitatea imaginii crește foarte mult dacă se folosesc mai multe raze pentru fiecare pixel. În figura 8 se consideră 4 raze care pleacă de la observator și trec prin colțurile zonei dreptunghiulare, din planul de proiecție, corespunzătoare unui pixel.

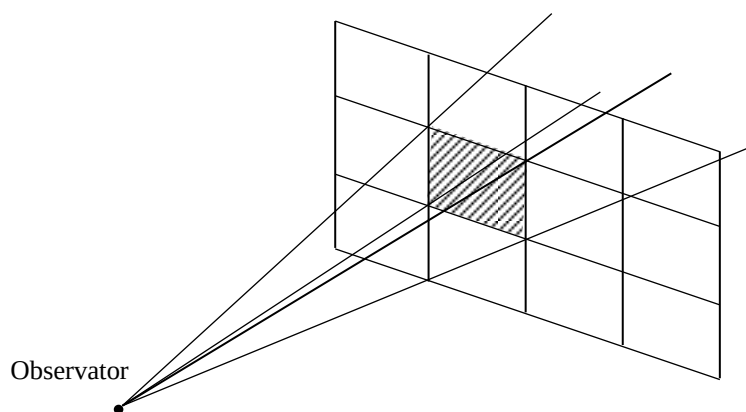


Fig. 8

Se observă de fapt că razele primare pentru un pixel sunt utilizate și la pixelii vecini. Dacă rezoluția imaginii este de $m \times n$ pixeli, sunt necesare $(m+1)(n+1)$ raze primare și nu $4 \times m \times n$.